

JEGYZŐKÖNYV

Mesterséges intelligencia és neurális hálózatok
Féléves beadandó

Készítette: **Orosz Kristóf**
Neptunkód: **EYZWG9**
Dátum: 2025. december 06.

Sárospatak, 2025

Tartalomjegyzék

Bevezetés	3
A projekt genetikus algoritmusa	3
HTML fájlok összefoglalója	5
CSS fájl összefoglalója	19
Adatbázis létrehozása	20
PHP fájlok összefoglalója	21

Bevezetés

A mesterséges intelligencia és neurális hálózatok című tantárgyból egy elektromos rollereket tároló depók készletkiegyenlítését készítettem el. A projektem során a genetikus algoritmus elvét használtam az optimális útvonal meghatározására, vagyis arra, hogy a tehergépkocsi a lehető legrövidebb úton tudja elosztani vagy összegyűjteni a rollereket a különböző depók között. A projekt front-end részét HTML és CSS segítségével valósítottam meg, amely a felhasználói felületet biztosítja. A JavaScript felel az útvonaloptimalizálási folyamat és a genetikus algoritmus logikai megvalósításáért. A PHP-fájlok az adatbázis-kezelést látják el, amelyek a MariaDB-ben létrehozott „MI_Beadando” adatbázissal kommunikálnak.

A projekt genetikus algoritmusa

A genetikus algoritmust az útvonaloptimalizálás problémájának megoldására használtam, ahol a cél a depók közötti legrövidebb útvonal megtalálása. A program kezdetben véletlenszerűen generál több lehetséges útvonalat (populációt), majd ezeken iterál generációkon keresztül. A szelkció során a rövidebb útvonalak előnyt élveznek, a keresztezés során a jó megoldások kombinálódnak, a mutáció pedig véletlenszerűen módosítja az útvonalakat a változatosság érdekében. minden generáció után a legjobb megoldások maradnak életben, így a rendszer fokozatosan egyre hatékonyabb útvonalakat talál. A folyamat 200 generáción keresztül ismétlődik, és a végén a legjobb azaz a legrövidebb útvonal kerül ki eredményként.

```
// Euklideszi távolság kiszámítása két pont között
function tavolsag(a, b) {
    return Math.sqrt((a.x - b.x) ** 2 + (a.y - b.y) ** 2);
}

// Egy útvonal teljes hossza
function utHossz(ut, raktarak) {
    let hossz = 0;
    for (let i = 0; i < ut.length - 1; i++) {
        const a = raktarak.find(d => d.nev === ut[i]);
        const b = raktarak.find(d => d.nev === ut[i + 1]);
        hossz += tavolsag(a, b);
    }
    // Visszatérés az induló raktárba
    const elseo = raktarak.find(d => d.nev === ut[0]);
    const utolso = raktarak.find(d => d.nev === ut[ut.length - 1]);
    hossz += tavolsag(utolso, elseo);
    return hossz;
}

// Populáció létrehozása (véletlen permutációk)
```

```
function letrehozPopulacio(raktarak, meret) {
    const populacio = [];
    const nevek = raktarak.map(r => r.nev);

    for (let i = 0; i < meret; i++) {
        const egyed = [...nevek];

        // Fisher-Yates keverés
        // Math floor függvény használata a genetikus algoritmus egész indexekre való
        kerekítése.

        // Max random függvény használata a véletlenszámok generálásához (pl. új populáció
        létrehozásához.)
        for (let j = egyed.length - 1; j > 0; j--) {
            const rand = Math.floor(Math.random() * (j + 1));
            [egyed[j], egyed[rand]] = [egyed[rand], egyed[j]];
        }
        populacio.push(egyed);
    }
    return populacio;
}

// Populáció rendezése útvonalhossz alapján
function rendez(pop, raktarak) {
    return pop.sort((a, b) => utHossz(a, raktarak) - utHossz(b, raktarak));
}

// Keresztezés
function keresztez(sz1, sz2) {
    const start = Math.floor(Math.random() * sz1.length);
    const end = Math.floor(Math.random() * sz1.length);

    // A Math min és Math max függvény amely során génerték korlátozása felső határhoz és
    génerték korlátozása alsó határhoz.
    const [min, max] = [Math.min(start, end), Math.max(start, end)];

    const resz = sz1.slice(min, max);
    const gyerek = sz2.filter(g => !resz.includes(g));
    return [...resz, ...gyerek];
}

// Mutáció (két elem felcserélése)
function mutacio(egyed, p) {
    const uj = [...egyed];
    if (Math.random() < p) {
        const i = Math.floor(Math.random() * uj.length);
        const j = Math.floor(Math.random() * uj.length);
        [uj[i], uj[j]] = [uj[j], uj[i]];
    }
    return uj;
```

```
}
```

```
// Genetikus algoritmus
function genetikusAlgoritmus(raktarak, generaciok = 200, popMeret = 60) {

    let populacio = letrehozPopulacio(raktarak, popMeret);

    for (let gen = 0; gen < generaciok; gen++) {

        // Szelekció: a populáció rendezése
        populacio = rendez(populacio, raktarak);

        const uj = [];

        // Populáció feltöltése keresztezéssel és mutációval
        while (uj.length < popMeret) {
            const sz1 = populacio[Math.floor(Math.random() * (popMeret / 2))];
            const sz2 = populacio[Math.floor(Math.random() * (popMeret / 2))];

            let gy = keresztez(sz1, sz2);
            gy = mutacio(gy, 0.2);

            uj.push(gy);
        }

        populacio = uj;
    }

    // Legjobb megoldás kiválasztása
    const legjobb = rendez(populacio, raktarak)[0];
    const tav = utHossz(legjobb, raktarak).toFixed(2);

    return { legjobb, tav };
}
```

HTML fájlok összefoglalója

Az index.html fájlt a projekt nyitóoldalaként készítettem el, amely az E-roller készletkiegyenlítő rendszer fő bemutató felülete. A felső menüsávban elhelyeztem a projekt címét, a saját azonosítómat, valamint két navigációs gombot, amelyek az útvonaltervezéshez és a rakodási naplóhoz vezetnek. A főoldal kétoszlopos elrendezésben jelenik meg: a bal oldalon a rendszer címe, a jobb oldalon pedig egy illusztrációs kép látható.

```
<!DOCTYPE html>
<html lang="hu">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>E-roller készletkiegyenlítő</title>
    <link rel="stylesheet" href="menu.css">

<style>
    /* A fő tartalmat középre igazító CSS */
    .tartalom-kontener {
        flex: 1;
        display: flex;
        justify-content: center;
        align-items: center;
        gap: 40px;
        flex-wrap: wrap;
        text-align: center;
        padding: 20px;
    }

    /* Bal oldali szövegoszlop stílusa */
    .bal-oszlop {
        flex: 1;
        min-width: 280px;
        max-width: 400px;
    }

    /* Címsor méretének beállítása */
    .bal-oszlop h1 {
        font-size: 1.8em;
        margin-bottom: 20px;
    }

    /* Jobb oldali képoszlop stílusa */
    .jobb-oszlop {
        flex: 1;
        min-width: 280px;
        max-width: 400px;
    }

    /* A főoldali e-roller kép formázása */
    .roller-kep {
        max-width: 100%;
        height: auto;
        border-radius: 10px;
    }

    /* Linkek egységes megjelenése */
```

```
a {  
    color: white;  
    text-decoration: none;  
}  
  
a:hover {  
    color: white;  
}  
  
/* Mobilnézethez: az elemek egymás alá kerülnek */  
@media (max-width: 768px) {  
    .tartalom-kontener {  
        flex-direction: column;  
    }  
}  
/  
</style>  
</head>  
  
<body>  
  
    <!-- Felső menüsáv -->  
    <header class="menu">  
        <div class="menu-bal">  
  
            <a href="index.html">  
                <b><i>E-roller készletkiegyenlítő</i></b>  
                <span style="font-size: 14px;"><br>Orosz Kristóf – EYZWG9</span>  
            </a>  
        </div>  
  
        <div class="menu-jobb">  
            <a href="utvonaltervezes.html"><button class="menu-gomb"><i>ÚTVONALTERVEZÉS</i></button></a>  
            <a href="rakodasinaplo.html"><button class="menu-gomb"><i>RAKODÁSI  
NAPLÓ</i></button></a>  
        </div>  
    </header>  
  
    <main class="tartalom-kontener">  
  
        <section class="bal-oszlop">  
            <h1><b><i>Városi közösségi e-roller készletkiegyenlítő rendszer</i></b></h1>  
        </section>  
  
        <section class="jobb-oszlop">  
              
        </section>  
    </main>  
}</body>
```

```
</section>
</main>

</body>
</html>
```

Az utvonaltervezes.html oldalt a rendszer adatbeviteli és előkészítő felületeként hoztam létre. Itt JavaScript segítségével lehetőség van új depóadat-blokkok dinamikus hozzáadására is, így rugalmasan bővíthető a lista. A depók adatai a depok.php fájlon keresztül érkeznek a MariaDB adatbázisból, majd a felhasználó által kiválasztott elemeket a rendszer a localStorage-ban tárolja. A „Optimalizálás indítása” gombbal a program a terkep.html oldalra irányítja a felhasználót.

```
<!DOCTYPE html>
<html lang="hu">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Útvonaltervezés</title>
    <link rel="stylesheet" href="menu.css">

    <style>
        .urlap-kontener {
            max-width: 600px;
            margin: 40px auto;
            background-color: #f9ffff;
            border: 2px solid #90EE90;
            border-radius: 10px;
            padding: 25px;
            color: #006400;
        }

        fieldset {
            margin-bottom: 20px;
            border: 2px solid #90EE90;
            border-radius: 8px;
            padding: 15px;
        }

        legend {
            font-weight: bold;
        }

        label {
            display: block;
            font-weight: bold;
        }
    </style>

```

```
    margin-top: 10px;
}

input, select {
    display: block;
    width: 100%;
    padding: 8px;
    margin-top: 5px;
    border: 1px solid #90EE90;
    border-radius: 4px;
    box-sizing: border-box;
}

.kuld-gomb {
    display: block;
    width: 100%;
    background-color: #006400;
    color: white;
    border: none;
    padding: 10px;
    border-radius: 5px;
    margin-top: 10px;
    font-weight: bold;
    cursor: pointer;
    transition: 0.3s;
}

.kuld-gomb:hover {
    background-color: #00a000;
}
</style>
</head>
<body>

<!-- Menü --&gt;
&lt;header class="menu"&gt;
    &lt;div class="menu-bal"&gt;
        &lt;a href="index.html" style="color:white;text-decoration:none;"&gt;
            &lt;b&gt;&lt;i&gt;E-roller készletkiegyenlítő&lt;/i&gt;&lt;br&gt;
                &lt;span style="font-size:14px;"&gt;Orosz Kristóf - EYZWG9&lt;/span&gt;&lt;/b&gt;
        &lt;/a&gt;
    &lt;/div&gt;

    &lt;div class="menu-jobb"&gt;
        &lt;a href="utvonaltervezes.html"&gt;&lt;button class="menu-
gomb"&gt;&lt;i&gt;ÚTVONALTERVEZÉS&lt;/i&gt;&lt;/button&gt;&lt;/a&gt;
        &lt;a href="rakodasinaplo.html"&gt;&lt;button class="menu-gomb"&gt;&lt;i&gt;RAKODÁSI
NAPLÓ&lt;/i&gt;&lt;/button&gt;&lt;/a&gt;
    &lt;/div&gt;</pre>
```

```
</header>

<!-- Tartalom -->
<main class="urlap-kontener">
  <h1><i>Útvonaltervezés</i></h1>
  <p>Kérem, válasszon depót az útvonal megtervezéséhez!</p>

  <fieldset id="depok">
    <legend>Depók adatai</legend>

    <div class="depo-blokk">
      <label>Depó neve:</label>
      <select class="depo-nev" onchange="betoltAdatok(this)">
        <option value="">-- Válassz depót --</option>
      </select>

      <label>X koordináta:</label>
      <input type="number" class="depo-x" readonly>

      <label>Y koordináta:</label>
      <input type="number" class="depo-y" readonly>

      <label>Aktuális készlet:</label>
      <input type="number" class="depo-aktualis" readonly>

      <label>Célkészlet:</label>
      <input type="number" class="depo-cel" readonly>
    </div>
  </fieldset>

  <button class="kuld-gomb" type="button" onclick="ujDepo()"+ Új depó
hozzáadása</button>
  <button class="kuld-gomb" type="button" onclick="adatokatKuldes()">Optimalizálás
indítása</button>
</main>

<script>
  let depoAdatok = [];

  // Depók betöltése az adatbázisból
  function betoltDepok() {
    fetch("depok.php")
      .then(response => response.json())
      .then(data => {
        depoAdatok = data;
        feltoltDepoLista();
      })
      .catch(err => console.error("Hiba a depók betöltésekor:", err));
  }
</script>
```

```
// Depók listájának frissítése
function feltoltDepoLista() {
    document.querySelectorAll(".depo-nev").forEach(select => {
        const elozo = select.value;
        select.innerHTML = "<option value='>-- Válassz depót --</option>";

        depoAdatok.forEach(depo => {
            const opcio = document.createElement("option");
            opcio.value = depo.nev;
            opcio.textContent = depo.nev;
            if (depo.nev === elozo) opcio.selected = true;
            select.appendChild(opcio);
        });
    });
}

// Új depó blokk hozzáadása
function ujDepo() {
    const szulo = document.getElementById("depok");
    const uj = document.createElement("div");
    uj.classList.add("depo-blokk");

    uj.innerHTML = `
        <label>Depó neve:</label>
        <select class="depo-nev" onchange="betoltAdatok(this)">
            <option value=""-- Válassz depót --</option>
        </select>

        <label>X koordináta:</label>
        <input type="number" class="depo-x" readonly>

        <label>Y koordináta:</label>
        <input type="number" class="depo-y" readonly>

        <label>Aktuális készlet:</label>
        <input type="number" class="depo-aktualis" readonly>

        <label>Célkészlet:</label>
        <input type="number" class="depo-cel" readonly>
    `;
    szulo.appendChild(uj);
    feltoltDepoLista();
}

// Kiválasztott depó adatainak betöltése
function betoltAdatok(selectElem) {
    const talalat = depoAdatok.find(d => d.nev === selectElem.value);
    if (talalat) {
```

```
const szulo = selectElem.closest(".depo-blokk");
szulo.querySelector(".depo-x").value = talalat.x;
szulo.querySelector(".depo-y").value = talalat.y;
szulo.querySelector(".depo-aktualis").value = talalat.aktualis;
szulo.querySelector(".depo-cel").value = talalat.cel;
}

}

// Adatok mentése localStorage-ba
function adatokatKuldes() {
    const depok = [];

    document.querySelectorAll(".depo-blokk").forEach(d => {
        depok.push(d.querySelector(".depo-nev").value);
    });

    localStorage.setItem("kiválasztottDepok", JSON.stringify(depok));
    window.location.href = "terkep.html";
}

window.onload = betoltDepok;
</script>

</body>
</html>
```

A terkep.html fájlt a depók térképi megjelenítésére és az útvonaloptimalizálás működésének vizuális szemléltetésére készítettem el. A kiválasztott depók a rácsos térképen jelennek meg, a helyüket az adatbázisban tárolt koordináták alapján határozzom meg. A genetikus algoritmust JavaScriptben valósítottam meg, amely több generáció keretében keresztül keresi a legrövidebb lehetséges útvonalat a depók között. Az optimalizálás eredményeként megjelenítem a legjobb útvonalat, valamint a hozzá tartozó távolságot.

```
<!DOCTYPE html>
<html lang="hu">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Rácsos térkép – E-roller készletkiegyenlítő</title>
    <link rel="stylesheet" href="menu.css">

    <style>
        /* A térképet megjelenítő 20x20-as rács stílusa */
        .racs-kontener {
```

```
display: grid;
grid-template-columns: repeat(20, 30px);
grid-template-rows: repeat(20, 30px);
gap: 2px;
margin: 40px auto;
background-color: #d9fcdb;
border: 2px solid #006400;
border-radius: 10px;
padding: 5px;
position: relative;
width: fit-content;
z-index: 1;
}

/* Egy-egy rácselem (cella) megjelenítése */
.cella {
    width: 30px;
    height: 30px;
    background-color: #fffffe;
    border: 1px solid #c2e5c2;
    border-radius: 4px;
    position: relative;
}

/* A depókat jelölő zöld körök formázása */
.raktar {
    background-color: #006400;
    border-radius: 50%;
    width: 22px;
    height: 22px;
    position: absolute;
    top: 4px;
    left: 4px;
    cursor: pointer;
    transition: transform 0.2s;
    z-index: 2;
}

/* Ha az egér a depó fölé kerül, nagyít és színt vált */
.raktar:hover {
    transform: scale(1.2);
    background-color: #00a000;
}

/* A kor (tooltip), amely a depó adatait mutatja */
.kor {
    position: absolute;
    background-color: rgba(0, 100, 0, 0.9);
    color: white;
```

```
padding: 4px 8px;
border-radius: 5px;
font-size: 12px;
white-space: nowrap;
transform: translate(-50%, -120%);
pointer-events: none;
opacity: 0;
transition: opacity 0.2s;
}

/* kor megjelenése egér fölé vitelekor */
.raktar:hover .kor {
  opacity: 1;
}

/* Az optimalizálás indítására szolgáló gomb */
.optimalizalas-gomb {
  margin: 15px auto;
  background-color: #006400;
  color: white;
  padding: 10px 20px;
  border: none;
  border-radius: 8px;
  cursor: pointer;
  font-weight: bold;
  font-size: 1em;
  transition: 0.3s;
  display: block;
  width: fit-content;
}

.optimalizalas-gomb:hover {
  background-color: #00a000;
}

/* Az eredmény megjelenítési doboz */
#utvonalEredmeny {
  margin: 20px auto;
  background: #f0ffff;
  border: 2px solid #90EE90;
  border-radius: 10px;
  padding: 15px;
  max-width: 600px;
  font-weight: bold;
  color: #004d00;
  white-space: pre-wrap;
  display: none;
}
```

```
/* Mobilbarát megjelenés */
@media (max-width: 768px) {
    .racs-kontener {
        grid-template-columns: repeat(10, 25px);
        grid-template-rows: repeat(10, 25px);
    }
}
</style>
</head>

<body>

<!-- Felső menüsáv az oldal navigációjához --&gt;
&lt;header class="menu"&gt;
    &lt;div class="menu-bal"&gt;
        &lt;a href="index.html" style="text-decoration: none; color: white;"&gt;
            &lt;b&gt;&lt;i&gt;E-roller készletkiegyenlítő&lt;/i&gt;&lt;br&gt;
                &lt;span style="font-size:12px;"&gt;Orosz Kristóf - EYZWG9&lt;/span&gt;&lt;/b&gt;
        &lt;/a&gt;
    &lt;/div&gt;
    &lt;div class="menu-jobb"&gt;
        &lt;a href="utvonaltervezes.html"&gt;&lt;button class="menu-gomb"&gt;&lt;i&gt;ÚTVONALTERVEZÉS&lt;/i&gt;&lt;/button&gt;&lt;/a&gt;
        &lt;a href="rakodasinaplo.html"&gt;&lt;button class="menu-gomb"&gt;&lt;i&gt;RAKODÁSI NAPLÓ&lt;/i&gt;&lt;/button&gt;&lt;/a&gt;
    &lt;/div&gt;
&lt;/header&gt;

<!-- A rácsos térkép megjelenítése --&gt;
&lt;div id="racs" class="racs-kontener"&gt;&lt;/div&gt;

<!-- Az optimalizálás indítása gomb --&gt;
&lt;button class="optimalizalas-gomb"&gt;Optimalizálás indítása&lt;/button&gt;

<!-- Az útvonal optimalizálás eredményének megjelenítése --&gt;
&lt;div id="utvonaleredmeny"&gt;&lt;/div&gt;

&lt;script&gt;
    // A kiválasztott depókat a localStorage-ból olvassa be
    const kivalasztottRaktarak = JSON.parse(localStorage.getItem("kivalasztottDepok") || "[]");

    // Ha nincs depó kiválasztva, visszairányít az utvonaltervezéshez
    if (kivalasztottRaktarak.length === 0) {
        alert("Nincsenek kiválasztott depók. Kérlek, térij vissza az útvonaltervezéshez!");
        window.location.href = "utvonaltervezes.html";
    }

    // A depók adatainak betöltése a PHP fájlból</pre>
```

```

fetch("depok.php")
  .then(valasz => valasz.json())
  .then(adatok => {
    const racs = document.getElementById("racs");

    // 20x20-as rács létrehozása
    for (let i = 0; i < 400; i++) {
      const cella = document.createElement("div");
      cella.classList.add("cella");
      racs.appendChild(cella);
    }

    // Kiválasztott raktárak megjelenítése a rácson
    adatok.forEach(raktar => {
      if (kivalasztottRaktarak.includes(raktar.nev)) {
        const x = parseInt(raktar.x);
        const y = parseInt(raktar.y);
        const index = y * 20 + x;
        const cella = racs.children[index];
        if (cella) {
          const raktarElem = document.createElement("div");
          raktarElem.classList.add("raktar");
          const kor = document.createElement("div");
          kor.classList.add("kor");
          kor.textContent = `${raktar.nev}\nKészlet: ${raktar.aktualis}/${raktar.cel}`;
          raktarElem.appendChild(kor);
          cella.appendChild(raktarElem);
        }
      }
    });
  });
</script>

<script>
// Euklideszi távolság kiszámítása két pont között
function tavolsag(a, b) {
  return Math.sqrt((a.x - b.x) ** 2 + (a.y - b.y) ** 2);
}

// Egy útvonal teljes hossza
function utHossz(ut, raktarak) {
  let hossz = 0;
  for (let i = 0; i < ut.length - 1; i++) {
    const a = raktarak.find(d => d.nev === ut[i]);
    const b = raktarak.find(d => d.nev === ut[i + 1]);
    hossz += tavolsag(a, b);
  }
  // Visszatérés az induló raktárba
  const also = raktarak.find(d => d.nev === ut[0]);

```

```

const utolso = raktarak.find(d => d.nev === ut[ut.length - 1]);
hossz += tavolsag(utolso, elso);
return hossz;
}

// Populáció létrehozása (véletlen permutációk)
function letrehozPopulacio(raktarak, meret) {
    const populacio = [];
    const nevek = raktarak.map(r => r.nev);

    for (let i = 0; i < meret; i++) {
        const egyed = [...nevek];

        // Fisher-Yates keverés
        for (let j = egyed.length - 1; j > 0; j--) {
            const rand = Math.floor(Math.random() * (j + 1));
            [egyed[j], egyed[rand]] = [egyed[rand], egyed[j]];
        }
        populacio.push(egyed);
    }
    return populacio;
}

// Populáció rendezése útvonalhossz alapján
function rendez(pop, raktarak) {
    return pop.sort((a, b) => utHossz(a, raktarak) - utHossz(b, raktarak));
}

// Keresztezés
function keresztez(sz1, sz2) {
    const start = Math.floor(Math.random() * sz1.length);
    const end = Math.floor(Math.random() * sz1.length);

    const [min, max] = [Math.min(start, end), Math.max(start, end)];

    const resz = sz1.slice(min, max);
    const gyerek = sz2.filter(g => !resz.includes(g));
    return [...resz, ...gyerek];
}

// Mutáció (két elem felcserélése)
function mutacio(egyed, p) {
    const uj = [...egyed];
    if (Math.random() < p) {
        const i = Math.floor(Math.random() * uj.length);
        const j = Math.floor(Math.random() * uj.length);
        [uj[i], uj[j]] = [uj[j], uj[i]];
    }
    return uj;
}

```

```
}
```

```
// Genetikus algoritmus
function genetikusAlgoritmus(raktarak, generaciok = 200, popMeret = 60) {

    let populacio = letrehozPopulacio(raktarak, popMeret);

    for (let gen = 0; gen < generaciok; gen++) {

        // Szelekció: a populáció rendezése
        populacio = rendez(populacio, raktarak);

        const uj = [];

        // Populáció feltöltése keresztezéssel és mutációval
        while (uj.length < popMeret) {
            const sz1 = populacio[Math.floor(Math.random() * (popMeret / 2))];
            const sz2 = populacio[Math.floor(Math.random() * (popMeret / 2))];

            let gy = keresztez(sz1, sz2);
            gy = mutacio(gy, 0.2);

            uj.push(gy);
        }

        populacio = uj;
    }

    // Legjobb megoldás kiválasztása
    const legjobb = rendez(populacio, raktarak)[0];
    const tav = utHossz(legjobb, raktarak).toFixed(2);

    return { legjobb, tav };
}

// A "Optimalizálás indítása" gomb eseménykezelője
document.querySelector(".optimalizalas-gomb").addEventListener("click", () => {
    fetch("depok.php")
        .then(res => res.json())
        .then(adatok => {
            const kivalasztottRaktarak = JSON.parse(localStorage.getItem("kivalasztottDepok"))
            || "[]";
            const aktivRaktarak = adatok.filter(d => kivalasztottRaktarak.includes(d.nev));
            const { legjobb, tav } = genetikusAlgoritmus(aktivRaktarak, 200);

            // Kiírja a legjobb útvonalat és a távolságot
            const eredmenyDiv = document.getElementById("utvonalEredmeny");
            eredmenyDiv.style.display = "block";
        })
});
```

```
        eredmenyDiv.innerHTML = `<b>Legjobb útvonal:</b><br>${legjobb.join(" →")}<br><br><b>Teljes távolság:</b> ${tav} egység`;

        utvonalRajzolas(legjobb, aktivRaktarak);
    });
});
</script>
</body>
</html>
```

CSS fájl összefoglalója

Ezzel a CSS-kód részlettel a weboldal kinézetét és elrendezését határoztam meg.

```
body {
    margin: 0;
    font-family: Arial, sans-serif;
    background-color: #90EE90;
    color: #006400;
    height: 100vh;
    display: flex;
    flex-direction: column;
}

.menu {
    display: flex;
    justify-content: space-between;
    align-items: center;
    background-color: green;
    color: white;
    padding: 15px 30px;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
}

.menu-bal {
    font-size: 1.2em;
    font-weight: bold;
}

.menu-gomb {
    background-color: transparent;
    border: 2px solid white;
    color: white;
    padding: 8px 16px;
    border-radius: 5px;
    cursor: pointer;
}
```

```
        font-weight: bold;
        transition: 0.3s;
    }

    .menu-gomb:hover {
        background-color: white;
        color: #006400;
    }

.tartalom-kontener {
    flex: 1;
    display: flex;
    justify-content: center;
    align-items: center;
    gap: 40px;
    flex-wrap: wrap;
    text-align: center;
    padding: 20px;
}

.bal-oszlop {
    flex: 1;
    min-width: 280px;
    max-width: 400px;
}

.bal-oszlop h1 {
    font-size: 1.8em;
    margin-bottom: 20px;
}

.jobb-oszlop {
    flex: 1;
    min-width: 280px;
    max-width: 400px;
}

@media (max-width: 768px) {
    .menu {
        flex-direction: column;
        text-align: center;
    }

    .menu-jobb {
        margin-top: 10px;
    }
}
```

Adatbázis létrehozása

Ebben a részben létrehoztam a projekt adatbázisát MariaDB-ben.

```
MariaDB [(none)]> CREATE DATABASE MI_beadando;
Query OK, 1 row affected (0.003 sec)
```

Létrehozom a depok című táblát.

```
MariaDB [(none)]> use MI_Beadando;
Database changed
MariaDB [MI_Beadando]> CREATE TABLE depok (
    ->     id INT AUTO_INCREMENT PRIMARY KEY,
    ->     nev VARCHAR(100) NOT NULL,
    ->     x INT NOT NULL,
    ->     y INT NOT NULL,
    ->     aktualis INT NOT NULL,
    ->     cel INT NOT NULL
    -> );
Query OK, 0 rows affected (0.024 sec)
```

Feltöltöm sárospataki utcanevekkel.

```
MariaDB [mi_beadando]> INSERT INTO depok (nev, x, y, aktualis, cel) VALUES
-> ('Rákóczi út, 3950 Sárospatak', 0, 0, 0, 12),
-> ('Comenius utca, 3950 Sárospatak', 1, 2, 0, 27),
-> ('Bartók Béla utca, 3950 Sárospatak', 2, 1, 0, 33),
-> ('Árpád utca, 3950 Sárospatak', 3, 4, 0, 44),
-> ('Szent Erzsébet utca, 3950 Sárospatak', 4, 3, 0, 58),
-> ('Kazinczy Ferenc utca, 3950 Sárospatak', 5, 1, 0, 63),
-> ('Dózsa György utca, 3950 Sárospatak', 6, 5, 0, 71),
-> ('Eőtvös József utca, 3950 Sárospatak', 7, 2, 0, 85),
-> ('József Attila utca, 3950 Sárospatak', 8, 6, 0, 90),
-> ('Hunyadi János utca, 3950 Sárospatak', 9, 7, 0, 99),
-> ('Váradí utca, 3950 Sárospatak', 10, 8, 0, 103),
-> ('Jókai Mór utca, 3950 Sárospatak', 11, 9, 0, 117),
-> ('Ady Endre utca, 3950 Sárospatak', 12, 10, 0, 120),
-> ('Kiss Ernő utca, 3950 Sárospatak', 13, 11, 0, 134),
-> ('Dobó István utca, 3950 Sárospatak', 14, 12, 0, 146),
-> ('Petőfi utca, 3950 Sárospatak', 15, 13, 0, 158),
-> ('Esze Tamás utca, 3950 Sárospatak', 14, 14, 0, 159),
-> ('Táncsics Mihály utca, 3950 Sárospatak', 13, 15, 0, 166),
-> ('Rózsa utca, 3950 Sárospatak', 12, 13, 0, 172),
-> ('Kossuth Lajos utca, 3950 Sárospatak', 11, 12, 0, 180),
-> ('Csokonai utca, 3950 Sárospatak', 10, 11, 0, 183),
-> ('Tinói utca, 3950 Sárospatak', 9, 10, 0, 187),
-> ('Bessenyei utca, 3950 Sárospatak', 8, 9, 0, 192),
-> ('Thököly utca, 3950 Sárospatak', 7, 8, 0, 195),
-> ('Vajda János utca, 3950 Sárospatak', 6, 7, 0, 199),
-> ('Lorántffy Zsuzsanna utca, 3950 Sárospatak', 5, 6, 0, 141),
-> ('Wesselényi utca, 3950 Sárospatak', 4, 7, 0, 54),
-> ('Szent Margit utca, 3950 Sárospatak', 3, 8, 0, 76),
-> ('Szígyet utca, 3950 Sárospatak', 2, 9, 0, 109),
```

PHP fájlok összefoglalója

Ebben a PHP-fájlban hozom létre az adatbázis-kapcsolatot a MI_Beadando névű adatbázissal.

```
<?php
// Az adatbázis-kapcsolat beállításához szükséges adatok megadása.
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "MI_Beadando";
```

```
// Létrehozom a kapcsolatot a MySQL adatbázissal a mysqli osztály segítségével.  
$conn = new mysqli($servername, $username, $password, $dbname);  
  
$conn->set_charset("utf8mb4");  
  
// Ellenőrzöm, hogy sikerült-e a kapcsolat.  
if ($conn->connect_error) {  
    die("Adatbázis kapcsolat sikertelen: " . $conn->connect_error);  
}  
?>
```

Ebben a PHP-fájlban a **depok** tábla adatait olvasom ki az adatbázisból, majd JSON formátumban adom vissza a kliensoldal számára.

```
<?php  
header('Content-Type: application/json; charset=utf-8');  
  
// Betöltöm az adatbázis-kapcsolatot a kapcsolat.php fájlból.  
require_once 'kapcsolat.php';  
  
// Lekérdezem az összes depót az adatbázisból a szükséges mezőkkel.  
$sql = "SELECT id, nev, x, y, aktualis, cel FROM depok";  
$result = $conn->query($sql);  
  
// Egy üres tömböt hozok létre, amibe a lekérdezett adatokat fogom eltárolni.  
$depok = [];  
  
// Ha a lekérdezés sikeres és van eredmény, bejárom a sorokat és beleteszem a tömbbe.  
if ($result && $result->num_rows > 0) {  
    while ($row = $result->fetch_assoc()) {  
        $depok[] = $row;  
    }  
    // A PHP tömböt JSON formátumba alakítom és visszaküldöm a kliensnek.  
    echo json_encode($depok, JSON_UNESCAPED_UNICODE | JSON_PRETTY_PRINT);  
} else {  
    // Ha nincs adat, hibaüzenetet küldök JSON formátumban.  
    echo json_encode(["error" => "Nem található adat a depok táblában."]);  
}  
  
$conn->close();  
?>
```